REMARKS

Claims 1-38 are pending in this application. By this Amendment, claims 2, 12, 16 and 33 are amended, with claims 2 and 12 amended for cosmetic purposes only and claims 16 and 33 amended to correct mistakes accidentally entered during prosecution.. The amendments are proper under 37 U.S.C. § 1.116 as the amendments merely comply with suggestions made by the Office Action. Reconsideration in view of the above amendments and the following remarks is respectfully requested.

The Office Action objects to claims 2, 12, 16-21 and 33. In response, claims 2, 12, 16 and 33 are amended to comply with the Office Action's suggestions. Accordingly, withdrawal of the objection is respectfully solicited.

The Office Action rejects claim 1-38 under 35 U.S.C. §103(a) over Iwamura et al. (U.S. Patent No. 5,945,976) in view of Montgomery et al. (U.S. Patent No. 5,969,533). This rejection is respectfully traversed.

In particular, Applicant asserts that it would not have been obvious at the time of the invention to modify Iwamura using the disclosure of Montgomery to teach or suggest a graphical user interface that includes a rendered image of at least one graphical object, wherein the graphical object uses a number of pixels on a display device and <u>a color value stored for each pixel in the display device</u>, as recited in independent claim 1, and similarly recited in independent claims 11, 22, 24, 29 and 33-36.

Iwamura discloses a graphic data processing system that displays a simulated *three-dimensional scene* from a number of different perspectives using on a vector map. See, Abstract and col. 1, lines 12-18. In an embodiment depicted in Fig. 5A and related text, the graphic data

processing system can be used to point to a "ground object" via an indication cursor 501. In operation of such an indication cursor 501 the graphic data processing system can use a number of approaches outlined in Figs. 9 and 10 and related text.

However, Iwamura makes no mention that any component of its graphics system uses either: (1) a color value stored for each pixel, or (2) object identification data stored with each pixel covered by a rendered image, issues that the Office Action again admits on page 3. Thus, Iwamura does not teach or suggest each and every limitation as recited in the independent claims.

Montgomery discloses a method for selecting an item from a *two-dimensional* graphics screen. See, Abstract and Fig. 2. As illustrated in Fig. 2, a number of graphic objects 204 and 206 can displayed with each object having: (1) an "item identifier" that identifies each particular graphics object to be displayed, and (2) a "color number" that is the sole instrument that defines the color of a given graphics object. See, col. 3, line 64 to col. 4, line 5. As clearly shown by Fig. 2 and related text, each object is represented by a single color. That is, there is but a single color available for each object in the Montgomery disclosure. Accordingly, Montgomery does not teach, suggest or even appreciate the use of a device that allows a separate color value stored for each pixel. Thus, Montgomery does not provide for the deficiencies of Iwamura.

Applicant respectfully asserts that the Office Action has misinterpreted the claim limitation "*a color value stored for each pixel in the display device.*" While the Office Action correctly states on page 7 that 'the word "separate" is not associated with "a color value of each pixel"', Applicants respectfully assert that the only interpretation feasible dictates a separate/respective color value for each pixel in a display device based both on the text of the specification and the plain meaning of the claim language.

For example, page 10, line 29 to page 11, line 2 recites: "As currently implemented, most computer screens 201 comprise a plurality of locations (e.g., "pixels") arranged in a two-dimensional array. Each pixel is on or off in a pure monochrome system, or activated with a red-green-blue (RGB) or cyan-magenta-yellow (CMY) or the like to <u>represent color on a pixel-by-pixel basis.</u>" {emphasis added}

Additionally, the Office Action's interpretation, as is exemplified on page 7 (e.g., "Montgomery has a pixel-by-pixel storage across an object's surface ...") indicates that the Office Action is attempting to interpret the language "*a color value stored for each pixel in the display device*" into meaning "*a color value stored for each pixel <u>of a graphical object</u> in the display device.*"

While Applicants do not contest that Montgomery stores a color value for each object, Applicant's do point out that, since not every pixel in the Montgomery device is necessarily covered by an object (see, e.g., Figure 7), Montgomery cannot not satisfy the limitation "*a color value stored for each pixel in the display device*" regardless of whether the Office Action attempts to expand the meaning of the claim language beyond what is clearly taught in the specification.

The Office Action has not provided any *prima facie* case of obviousness. To establish a *prima facie* case of obviousness, (1) the prior art references must teach or suggest all the claim limitations, (2) there must be some motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify or combine the reference teachings, and (3) there must be a reasonable likelihood of success that the claimed combination will work. See MPEP §2143, for example. *All three requirement must be met.*

As discussed above, Iwamura and Montgomery, individually or in combination, do not teach or suggest all the claim limitations.

Furthermore, there is no motivation, either in the cited references themselves or in the knowledge generally available to one of ordinary skill to modify Iwamura using the teachings of Montgomery. While the Office Action again states on pages 4 that 'it would have been obvious ... to use the item buffer technique of Montgomery to assign "identification data" to the scene image objects of Iwamura [in order] to provide the Iwamura user with a direct indexing to the identities of the contents of the scene image,' this stated motivation is problematic for a number of reasons.

First, the stated motivation is not found in any reference of record, and the Office Action provides no evidence, other than an unsupported conclusion, that this motivation would ever be apparent to one of ordinary skill in the art.

Second, it is not apparent that the virtual reality simulator of Iwamura would even benefit from using the "item buffer" described in Montgomery as compared to the present (z-buffering) system used by Iwamura, as such a modification would not be technically beneficial. That is, a z-buffering system modified using the item-buffer of Montgomery would cause the system to be slower and would introduce rendering errors.

Finally, a review of Iwamura reveals that *there can be NO reasonable likelihood of success* to modify the three-dimensional system of Iwamura to use Montgomery's item buffer, which is designed with two-dimensional graphics rendering in mind. That is, as with most three-dimensional rendering systems, Iwamura uses a "z-buffer" imaging approach to determine whether a particular object will be obfuscated by a second object (see, col. 8, line 53 of Iwamura), whereas the imaging approach used in Montgomery is limited by its "item buffer" technique to obfuscate based on order within a buffer, i.e., subsequent items listed obfuscate earlier items. See, Fig. 2 and col. 4, line 9+ of

Montgomery. Accordingly, the devices and methods of Montgomery are totally unsuitable for use with Iwamura, or in any system using modern three-dimensional graphics rendering.

While the Office Action replies to Applicants' earlier assertions (enumerated directly above) stating that "*it is not true ... since the resultant output of such rendering is a two dimensional image, whose pixels are then directly handled by a pixel-by-pixel arrangement such as Montgomery's*", Applicants first assert that the Office Action has not accounted for a fundamental aspect of z–buffering techniques that cannot be handled by Montgomery's item buffer.

That is, Montgomery's item buffer can not properly be used for a true three-dimensional scene where the perspective of a scene will change dynamically as: (1) Montgomery's item-buffer can only be used for two-dimensional objects (2) viewed from a single perspective.

Regarding issue (1), a review of Montgomery shows that only two-dimensional systems are described, and only two-dimensional objects, e.g., rectangles and triangles, are discussed. In fact, the term "three-dimensional" as well as any three-dimensional object, e.g. "cube" and "sphere", do not appear anywhere in the text of Montgomery.

Regarding issue (2), nowhere does Montgomery provide for any change in viewing perspective. For example, by changing a viewing perspective of a square from "head-on" to 60 degrees, a viewer could expect to see a rectangle (even ignoring shading, which is also not provided for by Montgomery). By further changing the viewing perspective by 90 degrees, a viewer would see a line. Where two separate two-dimensional objects are presented one in front of another, without depth information of each of the two objects (not provided for by Montgomery) it is impossible to determine the change in scene as a viewing perspective changes angularly, e.g, by rotation, or linearly, i.e., by zooming up to and even through one or both of the objects.

Once a z-buffering technique is performed, there is absolutely no subsequent use for any item-buffer as the z-buffering process already renders each pixel observable from a given perspective.

Further, even assuming that the Montgomery system was adapted to three-dimensions by arbitrarily assigning each point of each two-dimensional object three-dimensional coordinates, the process shown in Montgomery might be perhaps closest to a technique known as "*z-sorting*." The z-sorting algorithm simply displays all objects serially, starting with those objects furthest back (with the largest z-axis values). The z-sorting algorithm does not require a z-buffer, but it is slow and does not render intersecting objects correctly.[1]

Applicants respectfully assert that even a modest investigation will reveal that the z-buffering technique is one of the most efficient (if not the most efficient) three-dimensional rendering techniques known. Accordingly, Applicants suggest that any modification made to z-buffering (or the variant known as "w-buffering"), including that suggested by the Office Action, will result in a less efficient, slower and less capable system.

Thus, there can be no *prima facie* case of obviousness to modify Iwamura based on the teachings of Montgomery *as none of the requirements of a prima facie case of obviousness listed above has been met*, and indeed it appears that the suggested combination would result in a system totally unsuited for item selection in a three-dimensional graphics setting.

Thus, independent claims 1, 11, 22, 24, 29 and 33-38 are directed to patentable subject matter. The dependent claims are directed to patentable subject matter by virtue of their

[1] Applicant respectfully request the Examiner to carefully review z-buffering literature. Two websites currently available of interest (reproduced in the Appendix) include: http://www.webopedia.com/TERM/Z/Z_buffering.html and http://www.visionengineer.com/comp/z_buffering.shtml.

dependency as well as for the additional features they recite. Accordingly, Applicants respectfully request withdrawal of the rejection under 35 U.S.C. §103(a).

Applicants respectfully solicit that this Application is in condition for allowance, and Applicants request that the Examiner give the Application favorable consideration and permit it to issue as a patent. However, if the Examiner believes that the Application can be put in even better condition for allowance, the Examiner is invited to contact Applicants' representative listed below.

Please charge any shortage in fees due in connection with the filing of this paper, including extension of time fees, to Deposit Account 50-2036 and please credit any excess fees to such deposit account.

Respectfully submitted,

BAKER & HOSTETLER LLP

B. Y. Mathis
Registration No. 44,907

Date: September 9, 2005

Washington Square, Suite 1100
1050 Connecticut Avenue, NW
Washington, DC 20036
Phone: (202) 860-1500
Fax: (202) 861-1783

Attachments:
APPENDIX

APPENDIX

Z-buffering

Last modified: Wednesday, November 07, 2001

An algorithm used in 3-D graphics to determine which objects, or parts of objects, are visible and which are hidden behind other objects. With Z-buffering, the graphics processor stores the Z-axis value of each pixel in a special area of memory called the *Z-buffer* . Different objects can have the same x- and y-coordinate values, but with different z-coordinate values. The object with the lowest z-coordinate value is in front of

the other objects, and therefore that's the one that's displayed.

An alternate algorithm for hiding objects behind other objects is called *Z-sorting.* The Z-sorting algorithm simply displays all objects serially, starting with those objects furthest back (with the largest Z-axis values). The Z-sorting algorithm does not require a Z-buffer, but it is slow and does not render intersecting objects correctly.

•E-mail this definition to a colleague•

*For internet.com pages about **Z-buffering** CLICK HERE. Also check out the following links!*

LINKS = *Great Page!*

Related Categories

Object-Oriented Programming

Related Terms

3-D graphics

Z-buffer

JupiterWeb networks:

internet.com | EARTHWEB | dev | graphics.com

Search JupiterWeb: Find

Jupitermedia Corporation has three divisions:
JupiterImages, JupiterWeb and JupiterResearch

Jupitermedia Corporate Info | Newsletters | Tech Jobs | Shopping | E-mail Of

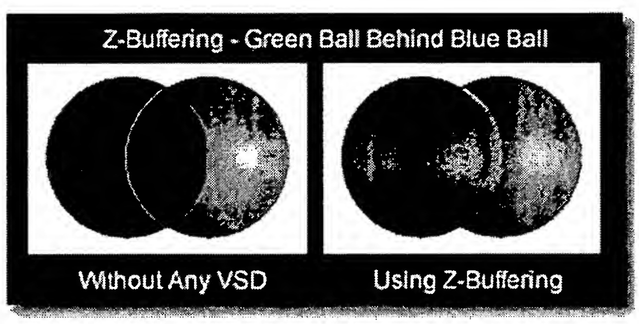
[an error occurred while processing this directive] [an error occurred while processing this directive] [an error occurred while processing this directive]

Z-Buffering

Article by : Duane Bong

Introduction To Z-Buffering

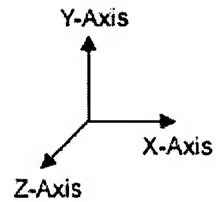
Z-buffering is a 3D imaging technique used for Visual Surface Determination [VSD]. In the real world, a solid object nearer to the viewer blocks and obscures other objects which are behind it. A quick example of this is to place your hand on the computer screen. Your hand blocks you from seeing any part of the screen that is behind it.



Z-Buffering Solves Overlapping Issues

However, when the same situation is generated in computer graphics, depth is lost and objects overlap each other. You will see both your hand and the image on screen at the same time! This is unavoidable because computer screens are flat and only 2 dimensional. To address this problem, Z-buffering is used to determine which objects are visible and which are hidden from view.

How Does Z-Buffering Work?



Before we proceed, let us examine a three dimensional Cartesian co-ordinate system. This has an X-axis, which defines left and right, and a Y-axis that defines up and down. It also has a Z-axis which tells us whether something will pop out of the screen or recede into it.

In Z-Buffering, the location of every pixel on the Z-axis is stored in memory. This is done for each individual object. The memory where the locations are stored is known as the Z-Buffer and this is what gives the technique its name. To decide on which objects should be displayed, the values in the Z-buffer are compared. An object with a lower Z-buffer value is considered to be in front of another object. The object with the lowest Z-Buffer value is, therefore, displayed, and the other objects hidden from view.

Z-Buffering is very good, in both static and animated scenes, at determining which objects need to be displayed. The technique痴 main disadvantage is that Z-Buffering needs to store and compare values of individual pixels. In many cases, this is a rather demanding load for the computer. Fortunately, most graphics cards are now equipped with specialised circuitry to speed up these demands.

Related Articles:

3D Graphics - Antialiasing Explained
3D Graphics - Texture Mapping Explained

3D Graphics - MIP Mapping Explained
3D Graphics - Alpha Blending Explained
3D Graphics - Bilinear Filtering Explained
3D Graphics - Bump Mapping Explained
3D Graphics - N-Patches Explained

Top

[an error occurred while processing this directive] [an error occurred while processing this directive]